SOFTWARE ENGINEERING PROCESSES AND MANAGEMENT

WORKSHOP 7
MEET THE METRICS

INTRODUCTION

The aim of this workshop is have you thinking about software metrics, their effectiveness, and what they really mean. Specifically, we will be discussing metrics for usability.

EFFECTIVE METRIC DESIGN

Recall the six following attributes for effective metrics:

1. Simple and computable.
2. Empirically and intuitively persuasive.
3. Consistent and objective.
4. Use of consistent units.
5. Programming language independent.
6. Useful for providing feedback.

A useful and effective metric need not have all of the above attributes, however, metrics that provide few of these are unlikely to be useful or effective.

USABILITY ENGINEERING

Usability engineering is a sub-field of software engineering that is concerned with the user-friendliness of interaction between humans and computers. The user-friendliness of a system is one of the most important factors in determining whether people will use it or not. For example, consider the range of devices released by Apple, such as the iPod and iPhone. These devices typically do little more than similar devices on the market — in some cases, much less! — but the user interfaces to these devices are outstanding, and help Apple to gain a large market share.

Unfortunately, despite being important, usability is also one of the most over-looked factors in software engineering. Designing a user interface is easy. Designing a *good* user interface is extremely difficult. For starters, how to we define "*good*"? To produce consistently user-friendly systems, we must turn to *usability engineering*.

Usability engineers typically use a range of tools and techniques to improve their products, such as questionnaires, cognitive walkthroughs, and usability testing. Like all aspects of software quality, usability cannot be put into a system after it has been developed[1]. It must be planned and designed into the system during the development process.

As a result, metrics for usability have been proposed by many researchers and practitioners. These metrics allow evaluation of usability before performing more expensive methods such as usability testing.

[1] Furthermore, the cost of changing the system at such a late stage is far greater than changing it at an earlier stage, such as during a requirements phases.

YOUR TASKS

Your tasks for the workshop are:

Class discussion (10 mins.) In your workshop groups, think of two systems or apps that you have used recently that you found easy to use, and two systems or apps that you found hard to use (yes, four in total). What was good about using the first two systems? Why were they usable? What was negative about the second two systems? What was it that made using them hard?

Metric design (20 mins.) Design *three* metrics for measuring *usability*. Discuss how you would use these metrics when engineering an application. Do your metrics have the attributes discussed in the section "Effective metric design" above? If not, justify.

Check your working (10 mins.) Consider a system that you have all used recently — perhaps a system identified in Task 1. Apply your metrics (as best you can) to this application. Evaluate the usability of the program using your metrics. How accurate are your metrics?

Linking back to basics (5 mins.) Do some of your metrics resemble other metrics discussed in the subject notes? If so, why do you think that is?